

# Integration of Real-World Graph Datasets into EasyGraph

July 2025

## 1 Introduction

This report documents the integration of eleven well-known graph datasets into the EasyGraph Python library. Each dataset was encapsulated in a dedicated class, with support for download, preprocessing, and loading into an EasyGraph-compatible format. The datasets cover a variety of domains and structures: citation networks, co-purchase graphs, social media interactions, road networks, and web hyperlinks.

Each dataset includes the following:

- A description and primary usage
- Dataset size (nodes, edges, features, classes)
- Directed or undirected nature
- What the dataset is best used for or famous for
- Reference of the original dataset
- Example usage output

## 2 Datasets

## 2.1 Amazon Photo

**Description:** A co-purchase graph of Amazon photo products.

**Use Case:** Node classification based on product categories.

**Famous For:** Semi-supervised learning benchmarks (used in PyG, DGL).

Size: 7,650 nodes, 119,082 edges, 745 features, 8 classes.

**Directed:** No

Dataset: <https://github.com/graphdml-uiuc-jlu/geom-gcn>

```
○ (venv) sana@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import AmazonPhotoDataset
dataset = AmazonPhotoDataset()
g = dataset[0]
print(g.number_of_nodes(), g.number_of_edges(), dataset.num_classes)
Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransform
Warning raise in module:model. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>> dataset = AmazonPhotoDataset()
Finished loading AmazonPhoto dataset.
NumNodes: 7650
NumEdges: 119082
NumFeats: 745
NumClasses: 8
Done saving data into cached files.
>>> g = dataset[0]
>>> print(g.number_of_nodes(), g.number_of_edges(), dataset.num_classes)
7650 119082 8
```

Figure 1: Amazon photos dataset

## 2.2 Arxiv HEP-TH

**Description:** Citation network of high-energy physics papers.

**Use Case:** Link prediction, network evolution.

**Famous For:** Studying densification laws (Leskovec et al.).

**Size:** 27,770 nodes, 352,807 edges.

**Directed:** Yes

Dataset: <https://snap.stanford.edu/data/cit-HepTh.html>

```
○ (venv) saman@node01:~/easyGraphPi/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import ArxivHEPThDataset
dataset = ArxivHEPThDataset()
G = dataset[0]

print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())
Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:models. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = ArxivHEPThDataset()
Downloading /users/sama/.EasyGraphData/cit-HepTh.txt.gz from https://snap.stanford.edu/data/cit-HepTh.txt.gz...
Finished loading Arxiv HEP-TH dataset.
  NumNodes: 27770
  NumEdges: 352807
Done saving data into cached files.
>>> G = dataset[0]
>>>
>>> print("Nodes:", G.number_of_nodes())
Nodes: 27770
>>> print("Edges:", G.number_of_edges())
Edges: 352807
>>>
>>>
```

Figure 2: Arxiv HEP-TH dataset

## 2.3 Coauthor CS

**Description:** Authors connected by co-authorship in computer science.

**Use Case:** Node classification and community detection.

**Famous For:** Used in Cluster-GCN and other scalable GNN benchmarks.

**Size:** 18,333 nodes, 81,894 edges, 6,805 features, 15 classes.

**Directed:** No

**Dataset:** [https://github.com/pyg-team/pytorch\\_geometric/blob/master/torch\\_geometric/datasets/coauthor.py](https://github.com/pyg-team/pytorch_geometric/blob/master/torch_geometric/datasets/coauthor.py)

```
py(venv) sama@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import CoauthorCSDataset
dataset = CoauthorCSDataset()
g = dataset[0]

print("Nodes:", g.number_of_nodes())
print("Edges:", g.number_of_edges())

node = list(g.nodes)[0]

print("Node 0 features shape:", g.nodes[node]["feat"].shape)
print("Node 0 label:", g.nodes[node]["label"])
print("Num classes:", dataset.num_classes) Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:model. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = CoauthorCSDataset()
Finished loading CoauthorCS dataset.
NumNodes: 18333
NumEdges: 81894
NumFeats: 6805
NumClasses: 15
Done saving data into cached files.
>>> g = dataset[0]
>>>
>>> print("Nodes:", g.number_of_nodes())
Nodes: 18333
>>> print("Edges:", g.number_of_edges())
Edges: 81894
>>>
>>> node = list(g.nodes)[0]
>>>
>>> print("Node 0 features shape:", g.nodes[node]["feat"].shape)
Node 0 features shape: (6805,)
>>> print("Num classes:", dataset.num_classes)
Num classes: 15
>>> 
```

Figure 3: Coauthor CS dataset

## 2.4 Facebook Ego

**Description:** Ego-centric friend networks on Facebook.

**Use Case:** Community detection, link prediction.

**Famous For:** Social circle prediction (SNAP EgoNets).

**Size:** 3,959 nodes, 84,243 edges.

Directed: No

Dataset: <https://snap.stanford.edu/data/ego-Facebook.html>

```
○ (venv) sam@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import FacebookEgoNetDataset
0]

print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())

Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:model. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = FacebookEgoNetDataset()
Finished loading Facebook Ego-Net dataset.
  NumNodes: 3959
  NumEdges: 84243
Done saving data into cached files.
>>> G = dataset[0]
>>>
>>> print("Nodes:", G.number_of_nodes())
Nodes: 3959
>>> print("Edges:", G.number_of_edges())
Edges: 84243
>>>
>>> █
```

Figure 4: Facebook ego dataset

## 2.5 Flickr

**Description:** Images on Flickr connected by common tags.

**Use Case:** Node classification using image features.

**Famous For:** GraphSAINT benchmark for sampling-based GNN training.

Size: 89,250 nodes, 449,878 edges, 500 features, 7 classes.

Directed: No

Dataset: <https://github.com/GraphSAINT/GraphSAINT>

```
○ (venv) sama@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import FlickrDataset
FlickrDataset(verbose=True)
g = ds[0]

print(g.number_of_nodes(), g.number_of_edges(), ds.num_classes)
print(len(g.graph["train_mask"]), sum(g.graph["train_mask"]))
Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:model. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> ds = FlickrDataset(verbose=True)
Downloading /users/sama/.EasyGraphData/flickr.zip from https://data.dgl.ai/dataset/flickr.zip...
Extracting file to /users/sama/.EasyGraphData/flickr
Loaded Flickr dataset
Nodes: 89250, Edges: 449878, Features: 500, Classes: 7
Done saving data into cached files.
>>> g = ds[0]
>>>
>>> print(g.number_of_nodes(), g.number_of_edges(), ds.num_classes)
89250 449878 7
>>> print(len(g.graph["train_mask"]), sum(g.graph["train_mask"]))
89250 44625
```

Figure 5: Flickr dataset

## 2.6 GitHub Users

**Description:** Social graph of GitHub users (followers).

**Use Case:** Node classification, influence modeling.

**Famous For:** Used in GraphSAINT and SNAP social graphs.

**Size:** 37,700 nodes, 289,003 edges, 5,575 features, 2 classes.

**Directed:** Yes

**Dataset:** <https://github.com/williamleif/GraphSAGE>

```
○ (venv) sama@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import GitHubUsersDataset

dataset = GitHubUsersDataset()
G = dataset[0]

print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())
print("Feature shape:", G.nodes[next(iter(G.nodes))]['feat'].shape)
print("Label:", G.nodes[next(iter(G.nodes))]['label'])

Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:models. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = GitHubUsersDataset()
Finished loading GitHub Users dataset.

NumNodes: 37700
NumEdges: 289003
Feature dim: 23
NumClasses: 2

Done saving data into cached files.
>>> G = dataset[0]
>>>
>>> print("Nodes:", G.number_of_nodes())
Nodes: 37700
>>> print("Edges:", G.number_of_edges())
Edges: 289003
>>> print("Feature shape:", G.nodes[next(iter(G.nodes))]['feat'].shape)
Feature shape: (19,)
>>> print("Label:", G.nodes[next(iter(G.nodes))]['label'])
Label: 0
>>> █
```

Figure 6: Github users dataset

## 2.7 Reddit (Posts)

**Description:** Reddit posts connected via comment/reply graph.

**Use Case:** Large-scale node classification by subreddit.

**Famous For:** Used in inductive GNN training (e.g., GraphSAGE).

**Size:** 232,965 nodes, 5,754,911 edges, 602 features, 41 classes.

**Directed:** No

**Dataset:** <https://github.com/williamleif/GraphSAGE>

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS python - Easy-Graph + □ 🗑

○ (venv) sama@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import RedditDataset
set = RedditDataset(self_loop=True)
g = dataset[0]

# Just check graph size and basic node attributes
print("Nodes:", g.number_of_nodes())
print("First node has label:", g.nodes[0]['label'])

Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:models. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = RedditDataset(self_loop=True)
Loaded Reddit dataset:
NumNodes: 232965
NumEdges: 57540911
NumFeats: 602
NumClasses: 41
```

Figure 7: Reddit posts dataset

## 2.8 RoadNet-CA

**Description:** Road network of California from intersections.

**Use Case:** Shortest paths, routing, connected components.

**Famous For:** Studying spatial networks and road connectivity.

**Size:** 1,965,206 nodes, 2,766,607 edges.

**Directed:** No

**Dataset:** <https://snap.stanford.edu/data/roadNet-CA.html>

```
○ (venv) sama@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import RoadNetCADataset
dataset = RoadNetCADataset()
G = dataset[0]

print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())
Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:model. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = RoadNetCADataset()
Downloading /users/sama/.EasyGraphData/roadNet-CA.txt.gz from https://snap.stanford.edu/data/roadNet-CA.txt.gz...
Finished loading RoadNet-CA dataset.
    NumNodes: 1965206
    NumEdges: 2766607
Done saving data into cached files.
>>> G = dataset[0]
>>>
>>> print("Nodes:", G.number_of_nodes())
Nodes: 1965206
>>> print("Edges:", G.number_of_edges())
Edges: 2766607
>>> █
```

Figure 8: RoadNet-CA dataset

## 2.9 Twitter Ego

**Description:** Twitter ego network built from user follows.

**Use Case:** Community analysis, influence tracking.

**Famous For:** Early SNAP ego-centric dataset.

**Size:** 81,306 nodes, 1,342,310 edges.

**Directed:** Yes

**Dataset:** <https://snap.stanford.edu/data/egonets-Twitter.html>

```
○ (venv) sama@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import TwitterEgoDataset
dataset = TwitterEgoDataset()
G = dataset[0] # __getitem__ will be triggered here

print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())
Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module:model. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = TwitterEgoDataset()
Done saving data into cached files.
>>> G = dataset[0] # __getitem__ will be triggered here
>>>
>>> print("Nodes:", G.number_of_nodes())
Nodes: 81306
>>> print("Edges:", G.number_of_edges())
Edges: 1342310
>>> █
```

Figure 9: Twitter ego dataset

## 2.10 WikiTopCats

**Description:** Directed graph of Wikipedia articles linked by top-level categories.  
**Use Case:** Multi-label classification, higher-order clustering, community detection.  
**Famous For:** Local clustering benchmarks, overlapping community research (KDD 2017).  
**Size:** 1,791,489 nodes, 28,511,807 edges.  
**Directed:** Yes  
**Dataset:** <https://snap.stanford.edu/data/wiki-topcats.html>

```
○ (venv) sams@node0:~/easyGraphPip/Easy-Graphs$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import WikiTopCatsDataset

ds = WikiTopCatsDataset()
G = ds[0]

print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())
print("Example node:", G.nodes[0]['name'])
print("Labels:", G.nodes[0]['label'][1:5])
Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning raise in nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning raise in module: model. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> ds = WikiTopCatsDataset()
Downloading /users/sams/_EasyGraphData/wiki-topcats.txt.gz from https://snap.stanford.edu/data/wiki-topcats.txt.gz...
Downloading /users/sams/_EasyGraphData/wiki-topcats-categories.txt.gz from https://snap.stanford.edu/data/wiki-topcats-categories.txt.gz...
Downloading /users/sams/_EasyGraphData/wiki-topcats-page-names.txt.gz from https://snap.stanford.edu/data/wiki-topcats-page-names.txt.gz...
Loaded graph: 1791489 nodes, 28511807 edges
Done saving data into cached files.
>>> G = ds[0]
>>>
>>> print("Nodes:", G.number_of_nodes())
Nodes: 1791489
>>> print("Edges:", G.number_of_edges())
Edges: 28511807
>>> print("Example node:", G.nodes[0]['name'])
Example node: 0 Chiasmal syndrome
>>> print("Labels:", G.nodes[0]['label'][1:5])
Labels: ['Category:Buterstroedeae', '301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396
397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448
449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605
606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657
658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762
763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814
815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 1791374]
>>>
>>>
>>> █
✖ SSH: and181.rah.cloudabu.su ① 0 △ 0 ② 0
```

Figure 10: WikiTopCats dataset

## 2.11 Web-Google

**Description:** Web pages and hyperlinks from Google's crawl.

**Use Case:** PageRank, centrality, structural analysis.

**Famous For:** SNAP web datasets, PageRank studies.

**Size:** 875,713 nodes, 5,105,039 edges.

**Directed:** Yes

**Dataset:** <https://snap.stanford.edu/data/web-Google.html>

```
○ (venv) sama@node0:~/easyGraphPip/Easy-Graph$ python
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from easygraph.datasets import WebGoogleDataset
>>> dataset = WebGoogleDataset()
G = dataset[0]

print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())
Please install Pytorch before use graph-related datasets and hypergraph-related datasets.
Warning: raise in module:nn. Please install Pytorch, torch_geometric, torch_scatter before you use functions related to AllDeepSet and AllSetTransformer.
Warning: raise in module:models. Please install Pytorch before you use hypergraph neural networks related to Hypergraph
>>>
>>> dataset = WebGoogleDataset()
Downloading /users/sama/.EasyGraphData/web-Google.txt.gz from https://snap.stanford.edu/data/web-Google.txt.gz...
Finished loading Web-Google dataset.
    NumNodes: 875713
    NumEdges: 5105039
    Done saving data into cached files.
>>> G = dataset[0]
>>>
>>> print("Nodes:", G.number_of_nodes())
Nodes: 875713
>>> print("Edges:", G.number_of_edges())
Edges: 5105039
>>> [REDACTED]
```

Figure 11: Web-Google dataset

## 3 Conclusion

The addition of these datasets significantly expands EasyGraph's built-in coverage, enabling rapid experimentation across domains. They support benchmarking of GNNs, topological algorithms, community detection, and more. All datasets follow a standardized interface with support for optional features, labels, and train/val/test masks when applicable.